Machine Learning in Healthcare – HW3

Yonathan Belicha | I.D: 203538574

1. Clustering

a. In my opinion K-medoid is more robust to outliers than K-means. While K-medoid uses only the samples from our dataset to find the best clustering (best medoid), K-means is trying to minimize the Euclidian metric and uses the mean value as the centroid. Thus, K-means algorithm most of the time will not use one of the dataset points as the centroid but will find a centroid that might not be exist on the dataset. If we have an outlier, K-means will try to minimize the distance by getting closer to the outlier, while K-medoid, that must use one of the dataset's points, will find the best cluster that exist and will be somewhere near the median value of the data.
For example, if our data looks like that: [-2, -1, 0, 1, 10000]. K-means will choose 1999.6 and the centroid, but as can be seen it is not a good cluster and not reflects a centroid that correlates with the real distribution of the data. K-medoid will run only on those 5 points and will try to find the minimal distance. Probably the point '0' will be chosen as the medoid.
Moreover, K-means has a start point and the clustering can be different in each time, depend on the initial condition. K-medoid is iterating all the dataset's points, and thus will converge to the same medoid each time without being depended on the initial condition (that can be an outlier in K-means). It might need more computational time in large dataset, but it is more reliable and less robust to outliers.

b. The problem as was declared as the question:

$$argmin \sum_{i=1}^{m} (x_i - \mu)^2$$

I will find the minimum by derivate the phrase and demand it will be equal to zero:

$$argmin \sum_{i=1}^{m} (x_i - \mu)^2 = \frac{\partial}{\partial \mu} \sum_{i=1}^{m} (x_i - \mu)^2 \textcolor{red}{=} \sum_{i=1}^{m} \frac{\partial}{\partial \mu} (x_i - \mu)^2 = \sum_{i=1}^{m} -2 \times (x_i - \mu) = 0$$

$$\sum_{i=1}^{m} 2 \times (\mu - x_i) = 0 \rightarrow 2\mu m - 2 \sum_{i=1}^{m} x_i = 0 \rightarrow \mu = \frac{1}{m} \sum_{i=1}^{m} x_i$$

- $\textcolor{red}{=}$ due to the linearity of the derivative.

There is a need to check if this point is a minimum, so I will check the second order derivative:

$$\frac{\partial}{\partial \mu} \sum_{i=1}^{m} -2 \times (x_i - \mu) \textcolor{red}{=} \sum_{i=1}^{m} \frac{\partial}{\partial \mu} - 2 \times (x_i - \mu) = 2 \times \sum_{i=1}^{m} \frac{\partial}{\partial \mu} (\mu - x_i) = 2 \times m > 0$$

Therefore, it is a minimum.

Bonus: I assume that x_k is the centroid, and thus: $x_k = \mu$.

Now, I will prove that $k = \frac{m}{2}$ and therefore, x_k is the also the median (assuming that m is even). The proof is based on that for indexes $i < k \rightarrow x_i < x_k$ and the also $i > k \rightarrow x_i > x_k$.

$$argmin \sum_{i=1}^{m} |x_i - \mu| = 0 \rightarrow \frac{\partial}{\partial \mu} \sum_{i=1}^{m} |x_i - \mu|$$

$$= \frac{\partial}{\partial \mu} \left(- \sum_{i=1}^{k-1} (x_i - \mu) + |x_k - \mu| + \sum_{i=k+1}^{m} (x_i - \mu) \right)$$

$$= \frac{\partial}{\partial \mu} \left(- \sum_{i=1}^{k-1} (x_i - \mu) + \sum_{i=k+1}^{m} (x_i - \mu) \right)$$

$$= \frac{\partial}{\partial \mu} \left(\mu \times (k-1) - \sum_{i=1}^{k-1} x_i - \mu \times (m-k-1) + \sum_{i=k+1}^{m} x_i \right)$$

$$= \frac{\partial}{\partial \mu} \left(\mu \times (2k-m) - \sum_{i=1}^{k-1} x_i + \sum_{i=k+1}^{m} x_i \right) = 2k - m = 0 \rightarrow k = \frac{m}{2}$$

Thus $x_{\frac{m}{2}} = \mu$, and therefore, $x_{\frac{m}{2}}$ is the centroid and the median. I proved that for the case that m is even, but if the dataset's size is odd $(m+1)$, it can be easily shown that the centroid will be at the dataset's median value, and therefore: $x_{\frac{m+1}{2}} = \mu$.

2. SVM

As can be seen, there are only 2 linear classifiers in the figures – A and D. Now there is a need to decide which of them is regarding to each of the linear SVM classifiers. In general, C is declared to be the punishment's value on misclassifying. The bigger C, thus there will be bigger punishment on misclassifying and therefore the margin will be less soft – more overfitting. If C is smaller, thus the margin will be smaller (soft margin) by allowing some points to be in the margin or be in the other side of the margin. Therefore, as can be seen, A have smaller margin that can be seen as two purple points located on the decision line (thus smaller C), and D has bigger margin (thus bigger C).

A – Linear kernel with $C = 0.01$.

D – Linear kernel with $C = 1$.

Now there is a need to decide which of the classifiers is polynomial. In general RBF uses gaussian model in infinite dimensional feature space to fit the model, thus it will tend to have margin that is more correlated to RBF kernel (more circular). So, as can be seen, C and F figures do not look like RBF classifier, so they belong to the polynomial classifier.

As learned, on the one hand, the 2nd degree polynomial classifier is more simple one and on the other hand the 10th degree polynomial classifier is more complex and might be overfitting. As can be seen easily, the F figure is overfitting by using too many "borders" (because its complexity and high polynomial degree), while C figure is more simple model.

C – 2nd order polynomial kernel.

F – 10th order polynomial kernel.

There are two figures that remain to classify: figures B and E. As said before, those two figures belong to the RBF classifier, so there is a need to understand which one belongs to each gamma.

In general, gamma is the hyperparameter that we use on RBF, and it detemines how much we fit to the training data. It declared as: $\gamma = \frac{1}{2\sigma^2}$. Therefore, the higher gamma, the model is more fitted to the training data. So, larger gamma can cause our model to be overfitted. Therefore, as can be seen, B figure is more overfitting and creates a close margin around the data (bigger gamma), while E figure is more flexible and not overfitting our data (smaller gamma).

E – RBF kernel with $\gamma = 0.2$.
B – RBF kernel with $\gamma = 1$.

3. ## Capability of generalization

a. The scientific term of the balance that Einstein meant in machine learning called – Generalization. As said in the lecture, generalization refers to how well the concepts learned by a machine learning model will translate to new observations not seen by the model when it was trained.[1] In machine learning, a good model that one creates is a model that is not too complex, and thus not overfitting ("Everything should be made as simple as possible…"). Also, the model should not be too simple, because it will not reflect the real data distribution ("but not too simple"). Those are the two major guidelines that one most pay attention when creating a good model – **those guidelines help to generalized well**.
In each classifier that one creates, there should be a declared cost function that one need to minimize and by that get a good model. In general, the cost function term should include a term that regulates the penalty for overfitting, also called regularization term, and a term that controls that our model will fit to the data. The balance between those two terms is regulated by using hyperparameters that can be estimated and changed according to the performances. Those hyperparameters control the tradeoff between too good fitting and too simple model. In general, by using a cost function, one can generalized well and create a good model that will be able to deal with unseen data.

b. In general, the model that will be chosen is the model that will minimize this kind of criterion – AIC. As can be seen '$2p$' term is in positive sign and declared to be the total number of learned parameters. So, bigger p indicates that there are more learned parameters, and therefore that our model correlates more to the data. Thus, this '$2p$' term becomes larger and the AIC term is become larger, so this model will not be chosen due to the reason of overfitting. '$2p$' term is kind of punishment term for overfitting.
Also can be seen that the '$2\ln(\hat{L})$' term is in negative sign and declared to be the estimated likelihood given these parameters. Because the natural logarithm is an increasing function, the larger $\hat{L}$ is (larger likelihood), the larger this term becomes and by that, AIC term becomes smaller and this model will be preferable. In contrast, if the $\hat{L}$ term is very small ($\hat{L} < 1$) the term '$2\ln(\hat{L})$' switch its sign and become positive, so the AIC becomes larger as the likelihood becomes smaller and therefore this model is not preferable.
Regarding 'a' section, in the AIC term there is a balance and a tradeoff between overfitting and likelihood to our training set. The preferable model will not be over fitted (small '$2p$' term) and will tend to have greater likelihood to fit the data (large '$2\ln(\hat{L})$' term). The model that has the best combination between those two terms will minimize AIC term and will be preferable because it generalized better.

c. As said in 'b' section, if this balance was violated, our model does not generalize well. It might be because of overfitting, due to too many learned parameters (bigger p) or might be due to choosing too simple model that does not fit well to our data (smaller $\hat{L}$). When one will try to apply the learned model on the test set or on real life data, he will get low scores and will see that his model did not generalize well.

d. AIC term is a term that should be smaller as can be. The smaller it gets; one can assume that his model does not overfit and has still has good correlation to the data. Small AIC term implies that the model can generalized well, reflects the reality, and will work well on unknown data. If this term is high, it means that the model has too many parameters or it is not fitting to our data – in such case the model that has large AIC is not preferable.

[1] BME-336546-Lecture-C06-Regularization – figure 5